PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Philippe Armangau, et al.	
Serial No.: 10/603,411 Confirm: 4172	Group Art Unit: 2168
Filed: 06/25/2003	Examiner: Oni, Olubusola
For: Replication of Snapshot Using a File System Copy Differential	Atty. Dkt. No.: EMCR:0095NPU Technology Center 2100

APPEAL BRIEF TO THE BOARD OF PATENT APPEALS AND INTERFERENCES

Commissioner for Patents
PO Box 1450
Alexandria, Virginia 22313-1450

Sir:

This Appeal Brief is in support of the Notice of Appeal filed Aug. 31, 2006 from the decision of the Examiner in the Final Official Action dated May 31, 2006. Please deduct any deficiency in any required fee from EMC Corporation Deposit Account No. 05-0889.

I. REAL PARTY IN INTEREST

The real party in interest is EMC Corporation, by virtue of an assignment recorded at Reel 014252 Frame 0875.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF THE CLAIMS

Claims 1-66 have been presented for examination.

Claims 1-7, 10-15, 19-32, 35-40, 44-53, 55-58, and 62-66 have been cancelled.

Claims 8-9, 16-18, 33-34, 41-43, 54, and 59-61 have been finally rejected, and are being appealed.

IV. STATUS OF AMENDMENTS

No amendment was filed after the final Official Action.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The appellants' invention of claim 8 provides a method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility receives a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies. The snapshot copy facility responds to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. (Appellants' specification, page 3, lines 4-11.) The snapshot copy facility has an index for each snapshot copy for indicating changes between said each snapshot copy and a next snapshot copy of the production file system. (Appellants' specification, page 3, lines 15-17.) For example, as shown in appellants' FIG. 25 and described in appellants' specification on page 62, lines 7-15, the index for each snapshot copy is a respective bit map of the snapshot copy facility of FIG. 25. The method of appellants' claim 8 includes scanning the indices for a sequence of the snapshot copies including the index for the specified older one of the snapshot copies and a respective index for each of a plurality of snapshot copies of the production file system that are both younger than the specified older one snapshot copies and older than the specified younger one of the snapshot copies. (Appellants' specification, page 3, lines 17-22.) For example, as shown in appellants' FIGS. 34 and 35, reproduced below, and described in the appellants' specification on page 63 lines 16-22, the indices for the sequence of the snapshot copies are scanned by a

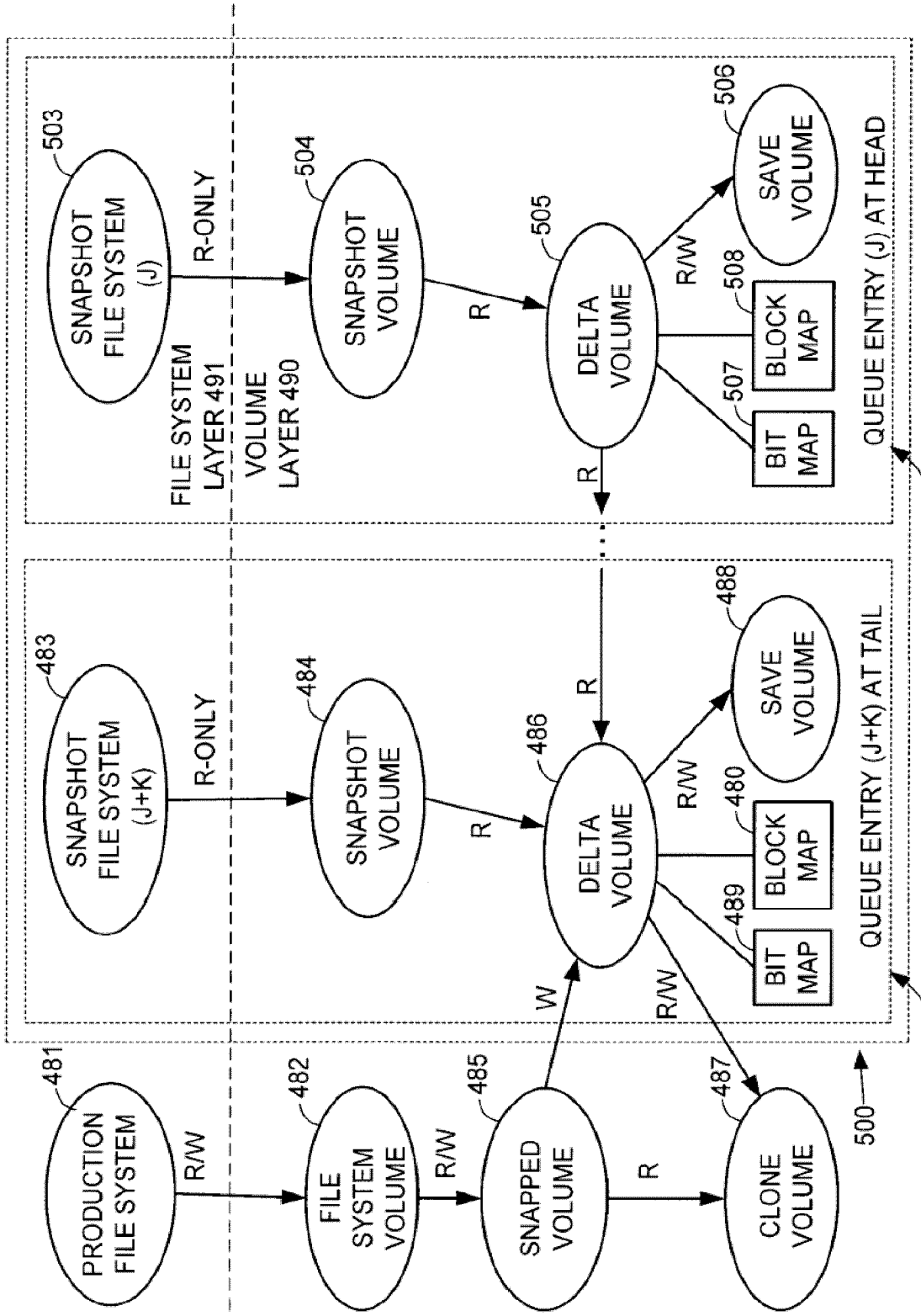

FIG. 25 (PRIOR ART)

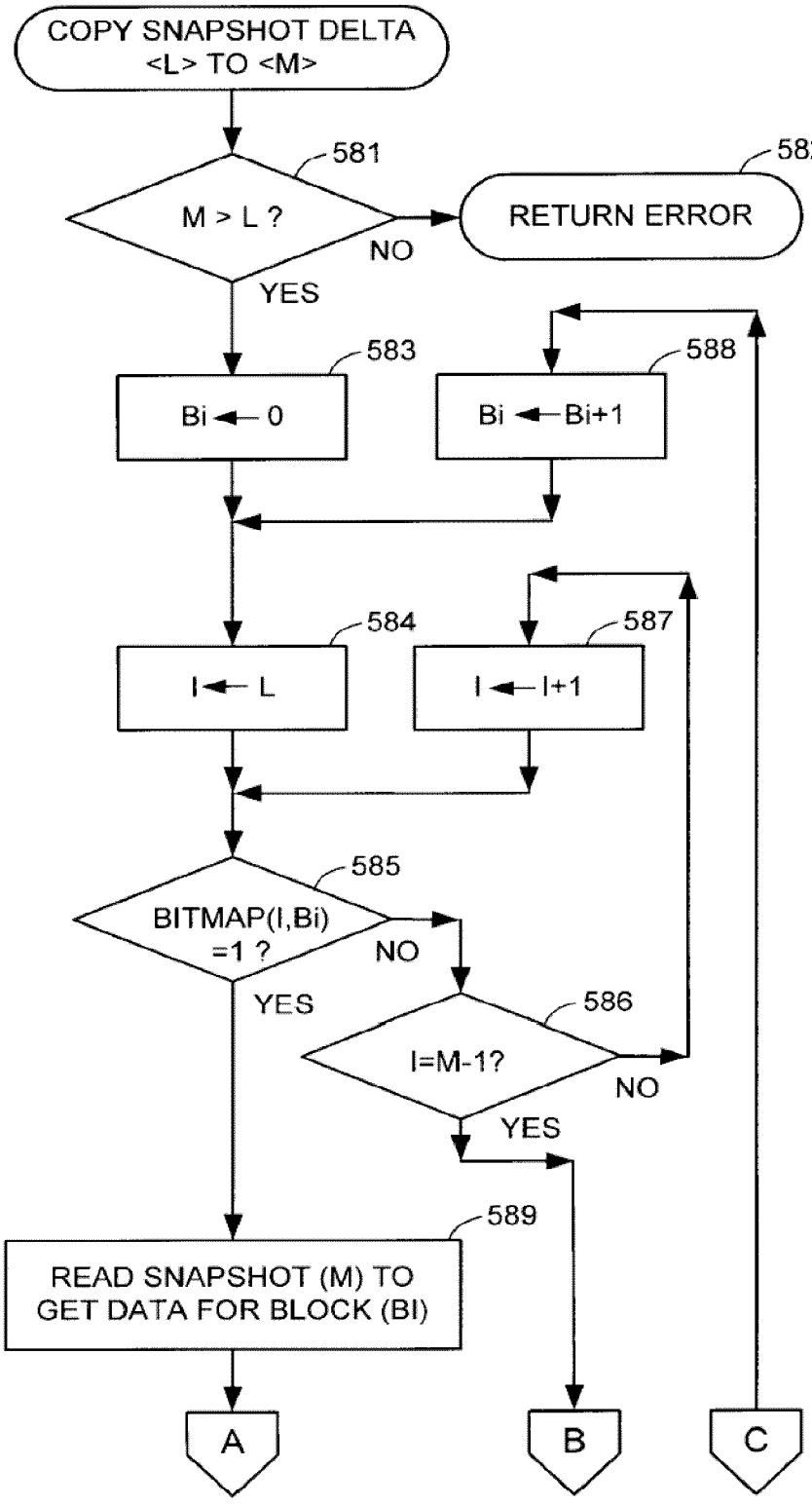
COPY SNAPSHOT DELTA <L> TO <M>
581
M > L ?
NO
582
RETURN ERROR
YES
583
Bi ← 0
588
Bi ← Bi+1
584
I ← L
587
I ← I+1
585
BITMAP(I,Bi) =1 ?
NO
YES
586
I=M-1?
NO
YES
589
READ SNAPSHOT (M) TO GET DATA FOR BLOCK (BI)
A
B
C

FIG. 34

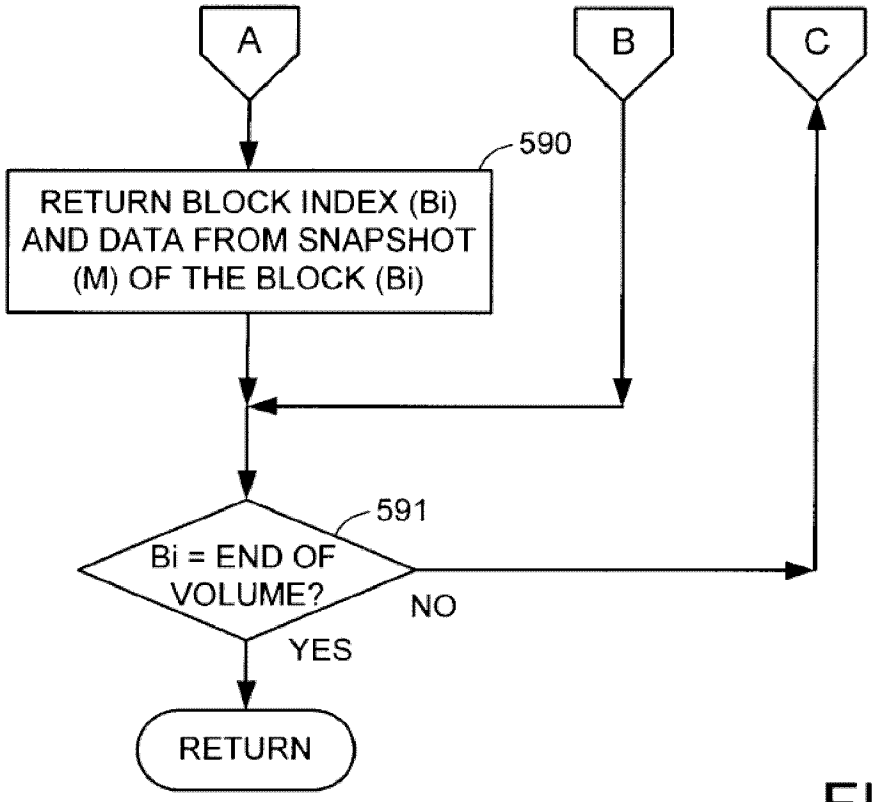

FIG. 35

program routine having an outer loop indexing blocks of data in the file system, and an inner loop indexing the snapshot copies in the sequence of the snapshot copies. "The program in the flowchart of FIGS. 34-35 has an inner loop including steps 585, 586, 587 that indexes the snapshots L to snapshot M-1. This sequence includes the snapshot L and the snapshots that are both younger than the snapshot L and older than the snapshot M. The program in the flowchart of FIGS. 34-35 has an outer loop including steps 584, 585, 586, 591, and 588 that indexes the blocks. When a bit in the indexed bit map is found to be set in step 585, the inner loop is exited to return the block index (B_i) and the data in the snapshot M for block (B_i)."

Appellants' claim 33 defines a snapshot copy facility including storage for storing a plurality of snapshot copies of a production file system, and at least one processor programmed for receiving a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies; and for responding to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies, using the method as specified in appellants' claim 8. See appellants' specification, page 5, line 25 to page 6, line 6; page 6 line 10 to line 18; appellants' FIG. 25 and appellants' specification on page 62, lines 7-15; and appellants' FIGS. 34-35 and appellant' specification, page 63, lines 11-22, as summarized above.

Appellants' claim 54 defines a program storage device containing a program for a snapshot copy facility, the snapshot copy facility storing a plurality of snapshot copies of a production file system, wherein the program is executable for responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies, using the method as specified in appellants' claim 8. See appellants' specification, page 8, lines 3-10 and 14-22; appellants' FIG. 25 and appellants' specification on page 62, lines 7-15; and appellants' FIGS. 34-35 and appellants' specification, page 63, lines 11-22, as summarized above.

Appellants' invention of claim 9 provides a method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy

facility receives a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies. The snapshot copy facility responds to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. (Appellants' specification, page 3, lines 4 to 11.) The snapshot copy facility has an index for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. For example, as shown in appellants' FIG. 42, as reproduced below, a snapshot copy facility has a meta bit map 691 for the production file system 481 and a respective meta bit map 692, 692 for each snapshot file system 483, 503) for each snapshot copy. (Appellants' specification, page 71, lines 12-14.) The meta bit map for the production file system has a bit for indicating whether or not each allocated block of storage in the production file system is valid or not. (Appellants' specification, page 70, lines 6-12.) Whenever a snapshot copy of the production file system is created, a snapshot copy of the meta bit map is also created. (Appellants' specification, page 71, lines 17-21.) The method of appellants' claim 9 further includes scanning the index for the specified younger one of the snapshot copies, and when the index indicates that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. The is shown in appellants' FIG. 40, as reproduced below, and as described in appellants' specification on page 68 line 11 to page

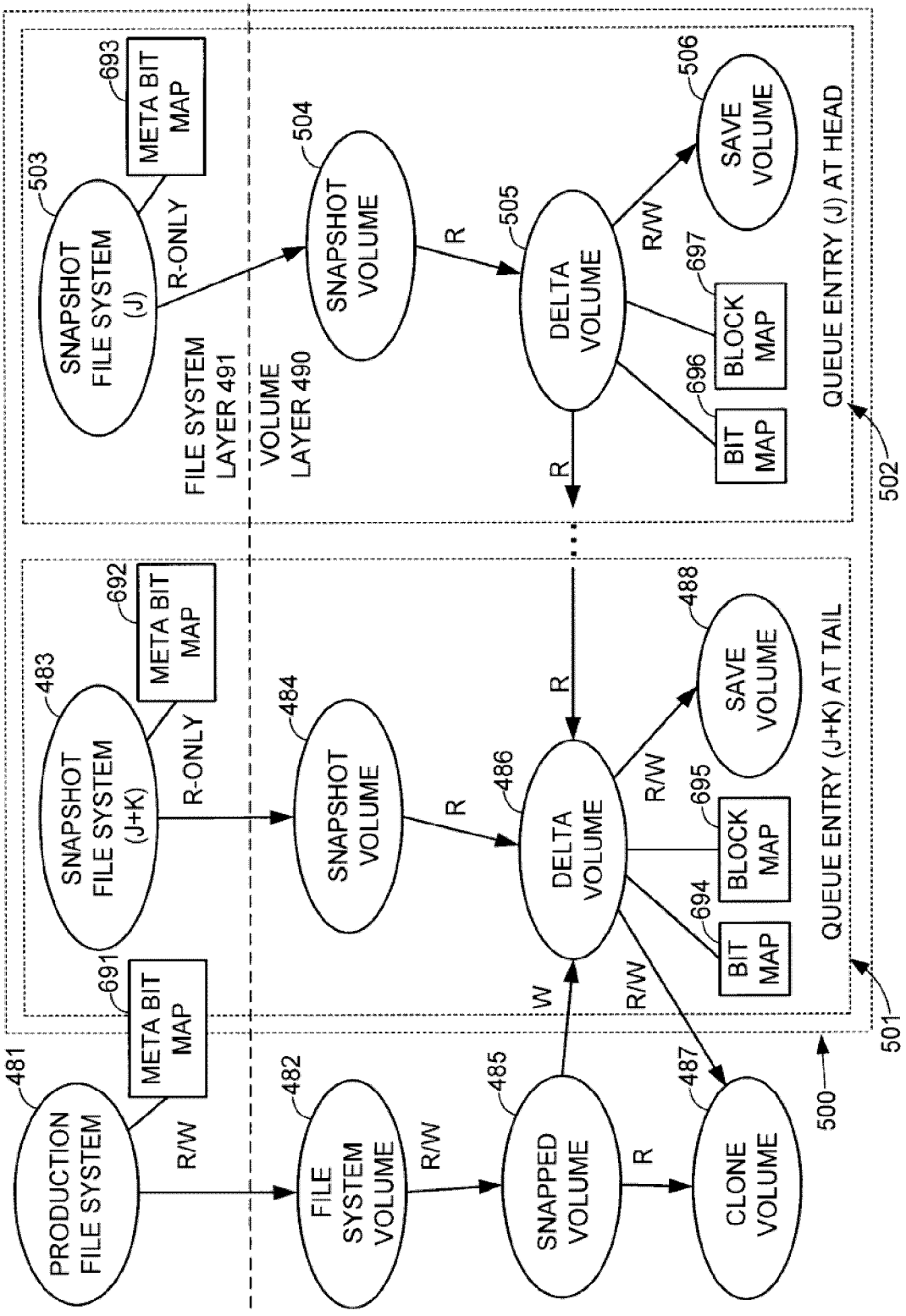
PRODUCTION FILE SYSTEM
481
META BIT MAP
691
R/W
FILE SYSTEM VOLUME
482
R/W
SNAPPED VOLUME
485
R
CLONE VOLUME
487
W
R/W
SNAPSHOT FILE SYSTEM (J+K)
483
META BIT MAP
692
R-ONLY
SNAPSHOT VOLUME
484
R
DELTA VOLUME
486
R
R/W
SAVE VOLUME
488
BLOCK MAP
695
BIT MAP
694
QUEUE ENTRY (J+K) AT TAIL
501
SNAPSHOT FILE SYSTEM (J)
503
META BIT MAP
693
R-ONLY
SNAPSHOT VOLUME
504
R
DELTA VOLUME
505
R
R/W
SAVE VOLUME
506
BLOCK MAP
697
BIT MAP
696
QUEUE ENTRY (J) AT HEAD
502
FILE SYSTEM LAYER 491
VOLUME LAYER 490
500

FIG. 42

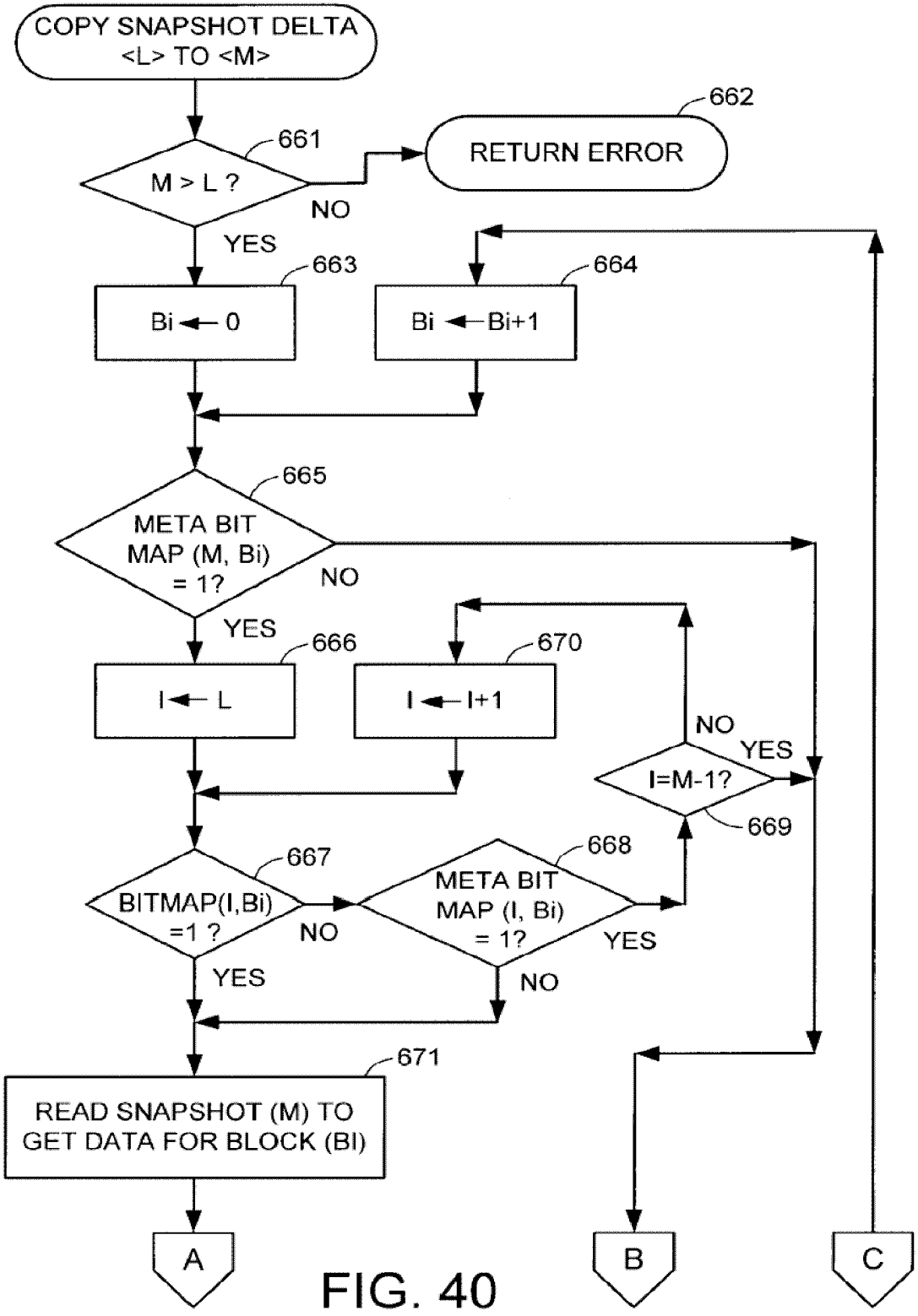
COPY SNAPSHOT DELTA <L> TO <M>
661
M > L ?
NO
662
RETURN ERROR
YES
663
Bi ← 0
664
Bi ← Bi+1
665
META BIT MAP (M, Bi) = 1?
NO
YES
666
I ← L
670
I ← I+1
NO
YES
I=M-1?
669
667
BITMAP(I,Bi) =1 ?
NO
668
META BIT MAP (I, Bi) = 1?
YES
YES
NO
671
READ SNAPSHOT (M) TO GET DATA FOR BLOCK (BI)
A
B
C

FIG. 40

69 line 5. "In step 665, the meta bit map for snapshot (M) [i.e., for the specified young one of the snapshot copies] has a value for block (B_i) indicating whether or not the block (B_i) is in use for the snapshot (M). In particular, a value of 1 indicates that the block (B_i) is in use for the snapshot (M)." (Appellants' specification, page 68, lines 14-15.) For a value of 1, execution continues to step 666 and then to step 667. "In step 667, if the bit map for snapshot (I) has a value of 1 for the block (Bi), then execution continues to step 671 to read the snapshot (M) to get data for the block (B_i) in order to return the data in response to the command to copy the snapshot delta <L> to <M>." (Appellants' specification, page 68, lines 17 to 20.) In other words, step 667 may determine that the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. The pertinent change for the block is read in step 471 from snapshot (M). The block index (Bi) and data from snapshot (M) of the block (Bi) are returned in step 590 of FIG. 35. (Appellants' specification, page 63, lines 2-4.)

Appellants' invention of claim 34 provides a snapshot copy facility. The snapshot copy facility includes storage for storing a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility also includes at least one processor programmed for receiving a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies; and for responding to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. (Appellants' specification, page 5, line 21 to page 6, line 6.) The snapshot

copy facility has an index for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. As summarized above with respect to claim 16, such a snapshot copy facility is shown in appellants' FIG. 42, and the index for each snapshot copy is a respective meta bit map (692, 693), as described in appellants' specification, page 71, lines 12-14; page 70, lines 6-12; and page 71, lines 17-21. The at least one processor is programmed for scanning the index for the specified younger one of the snapshot copies, and when the index indicates that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. Thus, the processor is programmed to perform the method of appellants' claim 9, as summarized above and as shown in appellants' FIG. 40 and described in appellants' specification page 68 line 11 to page 69 line 5.

Appellants' invention of claim 16 provides a method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility has an index for each snapshot copy for indicating blocks of data in the production file system that have changed between the snapshot copy and a next snapshot copy of the production file system. The method includes scanning the indices for a sequence of the snapshot copies to determine the blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies. The sequence of the snapshot copies includes the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies. (Appellants'

specification, page 3, lines 12 to 22.) The indices for the sequence of the snapshot copies are scanned by a program routine having an outer loop indexing respective blocks, and an inner loop indexing snapshot copies in the sequence of the snapshot copies. This is shown in appellants' FIG. 40, which is described on page 68 lines 11 to 16 of appellants' specification as including steps similar to steps in appellants' FIG. 39. The outer loop indexing respective blocks includes step 664, and the inner loop indexing the snapshot copies in the sequence of the snapshot copies includes step 670. The snapshot copy facility has a meta bit map for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. Such a snapshot copy facility is shown in appellants' FIG. 42 and described in appellants' specification, page 71, lines 12-14; page 70, lines 6-12; and page 71, lines 17-21, as summarized above for appellants' claim 9. The method further includes scanning the meta bit map for the specified younger one of the snapshot copies, and when the meta bit map is found to indicate that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies by scanning the indices for the sequence of the snapshot copies. This is shown in appellants' FIG. 40. As summarized above with respect to claim 9, the meta bit map is scanned in step 665 of FIG. 40, and when the meta bit map for the specified younger one (M) of the snapshot copies is a "1" indicating that a block is not known to be invalid, execution continues from step 665 to step 667 to then determine whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies by scanning the indices

(i.e., bitmaps in FIG. 42) in step 667 of FIG. 40 for the sequence of the snapshot copies, as described in appellants' specification on page 68 line 11 to page 69 line 5.

Appellants' invention of claim 41 provides a snapshot copy facility. The snapshot copy facility includes storage for storing a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility includes an index for each snapshot copy for indicating blocks of data in the production file system that have changed between the snapshot copy and a next snapshot copy of the production file system. The snapshot copy facility also includes at least one processor programmed for scanning the indices for a sequence of the snapshot copies to determine the blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies. (Appellants' specification, page 6, lines 7 to 18.) The snapshot copy facility includes a meta bit map for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. Such a snapshot copy facility is shown in appellants' FIG. 42 and described in appellants' specification, page 71, lines 12-14; page 70, lines 6-12; and page 71, lines 17-21, as summarized above for appellants' claim 9. The at least one processor is programmed for scanning the meta bit map for the specified younger one of the snapshot copies, and when the meta bit map is found to indicate that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of

the snapshot copies by scanning the indices for the sequence of the snapshot copies, as summarized above with respect to appellants' claim 16, and as shown in appellants' FIG. 40 and described in appellants' specification on page 68 line 11 to page 69 line 5.

Appellants' invention of claim 59 provides a program storage device containing a program for a snapshot copy facility. The snapshot copy facility has a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility also has an index for each snapshot copy for indicating blocks of data in the production file system that have changed between the snapshot copy and a next snapshot copy of the production file system. The program is executable for scanning the indices for a sequence of the snapshot copies to determine the blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies. (Appellants' specification, page 8, lines 11-22.) The snapshot copy facility has a meta bit map for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. Such a snapshot copy facility is shown in appellants' FIG. 42 and described in appellants' specification, page 71, lines 12-14; page 70, lines 6-12; and page 71, lines 17-21, as summarized above for appellants' claim 9. The program storage device is executable for scanning the meta bit map for the specified younger one of the snapshot copies, and when the meta bit map is found to indicate that a block is not known to be invalid, then determining whether the block has changed between the

specified older one of the snapshot copies and the specified younger one of the snapshot copies by scanning the indices for the sequence of the snapshot copies, as summarized above with respect to appellants' claim 16, and as shown in appellants' FIG. 40 and described in appellants' specification on page 68 line 11 to page 69 line 5.

Appellants' invention of claim 17 provides a method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility has a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between the snapshot copy and a next snapshot copy of the production file system and that have a "before image" saved for said each snapshot copy. The snapshot copy facility has a second index for each snapshot copy for indicating blocks of data that are not in use in the snapshot copy. The method includes responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by accessing the second index for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices for a sequence of the snapshot copies to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies. The sequence of the snapshot copies includes the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of

the snapshot copies. (Appellants' specification, page 4, lines 1-19.) Such a snapshot copy facility is shown in appellants' FIG. 42. The first index for each snapshot copy (483, 503) is a respective bit map (694, 696), and the second index for each snapshot copy is a respective meta bit map (692, 693). The bit map is described in appellants' specification on page 62, lines 7-15. The meta bit map is described in appellants' specification on page 66, lines 10-12: "In a preferred snapshot copy facility, as described below with reference to FIGS. 41 to 46, there is kept a meta bit map for each snapshot copy for indicating blocks of the production file system that are not in used in the snapshot copy." FIGS. 39 and 40 show an example of responding to a request for the difference between a specified older one (L) of the snapshot copies and a specified younger one (M) of the snapshot copies by accessing the second index (meta bit map) in step 665 of FIG 40) for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies (see corresponding step 655 in FIG. 39), and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices (in step 667 of FIG. 40) for a sequence of the snapshot copies (e.g., in the inner loop of steps 667, 668, 669, and 670 in FIG. 40) to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, as described in appellants' specification on page 67 line 11 to page 69 line 5.

Appellants' invention of claim 42 provides a snapshot copy facility. The snapshot copy facility includes storage for storing a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in

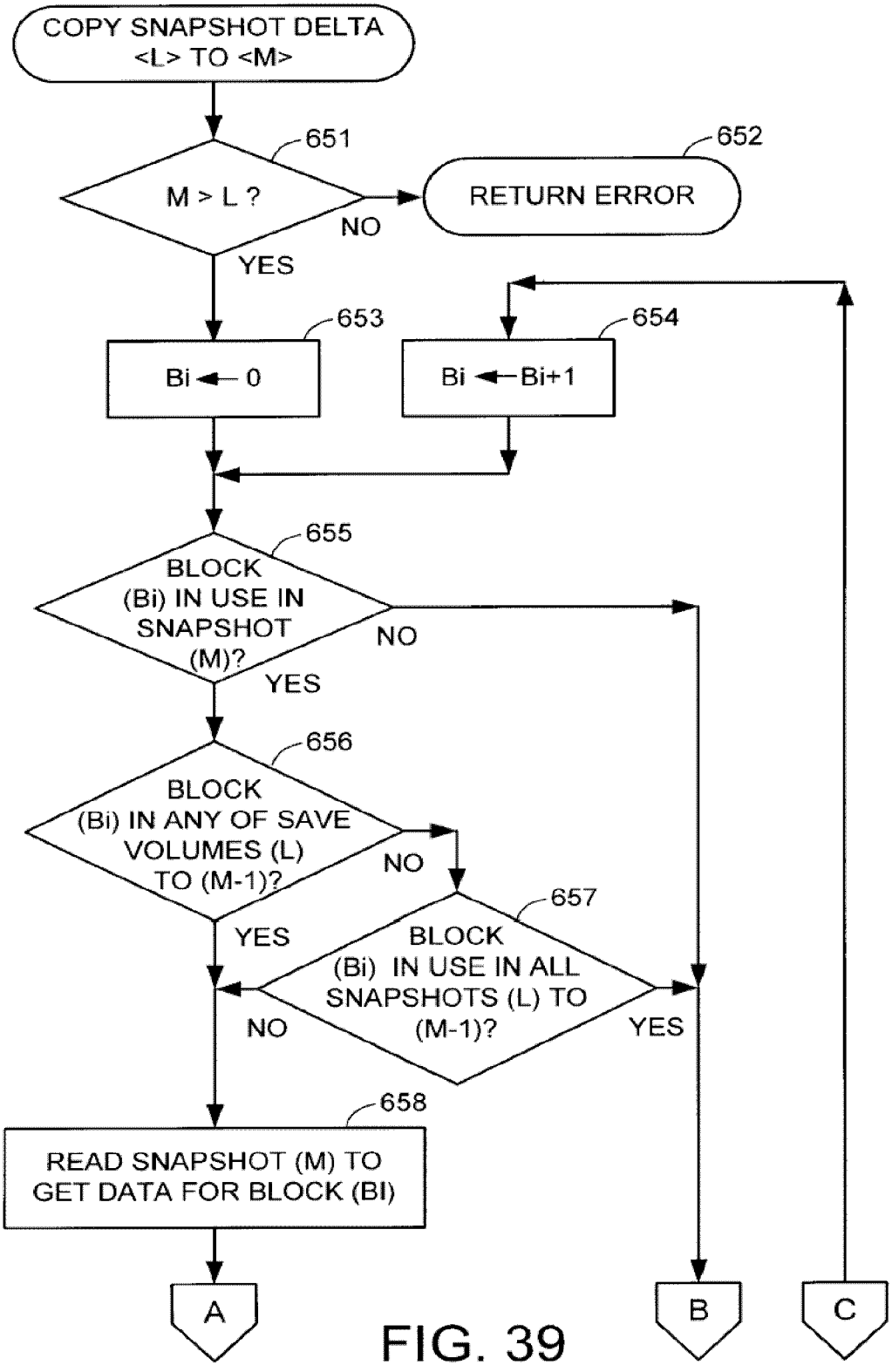
COPY SNAPSHOT DELTA <L> TO <M>
651
M > L ?
NO
652
RETURN ERROR
YES
653
Bi ← 0
654
Bi ← Bi+1
655
BLOCK (Bi) IN USE IN SNAPSHOT (M)?
NO
YES
656
BLOCK (Bi) IN ANY OF SAVE VOLUMES (L) TO (M-1)?
NO
YES
657
BLOCK (Bi) IN USE IN ALL SNAPSHOTS (L) TO (M-1)?
NO
YES
658
READ SNAPSHOT (M) TO GET DATA FOR BLOCK (BI)
A
B
C

FIG. 39

The snapshot copy facility has a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between the snapshot copy and a next snapshot copy of the production file system and that have a "before image" for the snapshot copy stored in the storage. The snapshot copy facility has a second index for each snapshot copy for indicating blocks of data that are not in use in the snapshot copy. The snapshot copy facility also has at least one processor programmed for responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by accessing the second index for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices for a sequence of the snapshot copies to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies. The sequence of the snapshot copies includes the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies. (Appellants' specification, page 6, line 19 to page 7, line 15.) Such a snapshot copy facility is shown in appellants' FIG. 42. The first index for each snapshot copy (483, 503) is a respective bit map (694, 696), and the second index for each snapshot copy is a respective meta bit map (692, 693). FIGS. 39 and 40 show an example of how the first index and the second index are used, as described in the appellants' specification and as summarized above with respect to appellants' claim 17.

The invention of appellants' claim 60 provides a program storage device containing a program for a snapshot copy facility. The snapshot copy facility has a plurality of snapshot copies of a production file system. Each of the snapshot copies is a prior state of the production file system at a respective point in time. The snapshot copy facility has a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between the snapshot copy and a next snapshot copy of the production file system and that have a "before image" for the snapshot copy stored in the snapshot copy facility. The snapshot copy facility has a second index for each snapshot copy for indicating blocks of data that are not in use in the snapshot copy. The program is executable for responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by accessing the second index for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices for a sequence of the snapshot copies to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies. The sequence of the snapshot copies includes the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies. (Appellants' specification, page 9, lines 1-20.) Such a snapshot copy facility is shown in appellants' FIG. 42. The first index for each snapshot copy (483, 503) is a respective bit map (694, 696), and the second index for each snapshot copy is a respective meta bit map (692, 693).

Appellants' FIGS. 39 and 40 show an example of how a program uses the first index and the second index, as described in the appellants' specification and as summarized above with respect to appellants' claim 17.

Claims 18, 43, and 61 are dependent upon claims 17, 42, and 60, respectively. Each of claims 18, 43, and 61 recites "accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the sequence of the snapshot copies to determine that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies not changed." Determining that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies occurs in steps 656 and 657 of FIG. 39, and more specifically in the inner loop of steps 667, 668, 669, and 670 in FIG. 40. The "accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the sequence of the snapshot copies to determine that said at least one of the blocks has changed" occurs in step 668 of FIG. 40 when the meta bit map (I, Bi) is not equal to one, causing execution to branch from step 668 to step 671 in FIG. 40. As described in appellants' specification, page 68 line 22 to page 69 line 4: "In step 668, if the meta bit map for the snapshot (I) does not have a value of 1 for the block (B_i), execution continues to step 671 to read the snapshot (M) to get data for the block (B_i), in order to return the data in response to the command to copy the snapshot delta <L> to <M>. In this case, the block (B_i) is not in use in the snapshot (I)."

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Whether claims 8, 33, and 54 are unpatentable under 35 U.S.C. 102(e) as being anticipated by Goldstein et al. (Pub. No. US 20040163009).

2. Whether claims 9, 16, 17, 18, 34, 41, 42, 43, 59, 60, and 61 are unpatentable under 35 U.S.C. 103(a) over Goldstein et al. (Pub. No. US 20040163009) in view of Ohran et al. (Pub. No. US 20020112134).

VII. ARGUMENT

1. Claims 8, 33, and 54 are not unpatentable under 35 U.S.C. 102(e) over Goldstein et al. (Pub. No. US 20040163009).

"For a prior art reference to anticipate in terms of 35 U.S.C. § 102, every element of the claimed invention must be identically shown in a single reference." Diversitech Corp. v. Century Steps, Inc., 7 U.S.P.Q.2d 1315, 1317 (Fed. Cir. 1988), quoted in In re Bond, 910 F.2d 831,15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990) (vacating and remanding Board holding of anticipation; the elements must be arranged in the reference as in the claim under review, although this is not an ipsis verbis test).

Goldstein discloses a backup apparatus and method suitable for protecting the data volume in a computer system by acquiring a base state snapshot and a sequential series of data volume snapshots. The apparatus concurrently generates succedent and precedent lists of snapshot differences which are used to create succedent and precedent backups respectively. The data volume is restored by overwriting the base state data with data blocks identified in one or more succedent backups. File recovery is accomplished by overwriting data from a concurrent snapshot with one or more precedent backups. (Goldstein, Abstract.)

Goldstein teaches that blocks that have change between snapshots are identified in a list, and are copied into backups and stored in offline storage. (Page 2, paragraph [0024].) A first

succedent backup 131 (B01) is created by copying from the first state snapshot 113 (S1) all of the data blocks identified in the first succedent snapshot difference list 121. A copy of the snapshot difference list 121 is also included in the first succedent backup 131. (Page 2, paragraph [0030].) Successive precedent backups may be combined into a single precedent backup to reduce offline storage volume and to speed incremental file recovery, as shown in FIG. 10. (Page 4, paragraph [0050].)

In appellants' amendment filed March 23, 2006, claims 8, 33, and 54 were amended to clearly distinguish Goldstein by defining that the sequence of the snapshot copies that is scanned includes the index for the specified older one of the snapshot copies and a respective index for each of a plurality of snapshot copies of the production file system that are both younger than the specified older one snapshot copies and older than the specified younger one of the snapshot copies. Thus, the indices for this particular sequence of the snapshot copies are scanned by a program routine having an outer loop indexing blocks of data in the file system, and an inner loop indexing the snapshot copies in the sequence of the snapshot copies. This is shown in appellants' FIGS. 34 and 35, for example, and described in the appellants' specification on page 63 lines 16-22.

Disclosure in Goldstein pertinent to appellants' claims 8, 33, and 54 is the generation of concatenated precedent lists for example as recited in Goldstein's claim 10 on page 6. The appellants' amendment to claims 8, 33, and 54 clearly define a scanning procedure including inner and outer loops that determine the blocks that have changed over a series of at least three successive snapshots. Instead of generating concatenated precedent snapshot difference lists

between neighboring snapshots in a series by a process of repeated concatenation (e.g., indexing blocks in an inner loop and indexing snapshot copies in an outer loop), the appellants' claims define indexing the snapshot copies in an inner loop and indexing blocks in the outer loop. The appellants' claims call for a scanning procedure directly opposite to what Goldstein would suggest.

In reply to the appellants' amendment and argument, the final Official Action on page 11 cites Goldstein paragraphs [0011], [0024-0033], [0041-0043], and fig. 3, 4, 6 & 7, and concludes "Wherein Goldstein's teachings involve acquiring snapshots of consistent states of data volume, wherein the snapshots are compared to produce a list of blocks that have changed between the snapshots, i.e., the determination of blocks that have changed, thus teaches are synonymous)." However, anticipation under 35 U.S.C. 102 is not satisfied by a teaching of a different way of how to determine blocks that have changed. Instead, for anticipation under 35 U.S.C. 102, every element of the claimed invention must be identically shown in a single reference. See Diversitech Corp. and In re Bond cited above.

2. Claims 9, 16, 17, 18, 34, 41, 42, 43, 59, 60, and 61 are not unpatentable under 35 U.S.C. 103(a) over Goldstein et al. (Pub. No. US 20040163009) in view of Ohran et al. (Pub. No. US 20020112134).

The policy of the Patent and Trademark Office has been to follow in each and every case the standard of patentability enunciated by the Supreme Court in Graham v. John Deere Co., 148 U.S.P.Q. 459 (1966). M.P.E.P. § 2141. As stated by the Supreme Court:

> Under § 103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter is determined. Such secondary considerations as commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented. As indicia of obviousness or nonobviousness, these inquiries may have relevancy.

148 U.S.P.Q. at 467.

The problem that the inventor is trying to solve must be considered in determining whether or not the invention would have been obvious. The invention as a whole embraces the structure, properties and problems it solves. In re Wright, 848 F.2d 1216, 1219, 6 U.S.P.Q.2d 1959, 1961 (Fed. Cir. 1988).

It is improper for the Examiner to rely on an applicant's novel discoveries or teachings in his or her own patent application to establish that the invention claimed in that patent application would have been obvious. Riverwood International Corp. v. R.A. Jones & Co., 324 F.3d 1346, 1355, 66 U.S.P.Q.2d 1331, 1338 (Fed. Cir. 2003)("Ones own work may not be considered prior art in the absence of a statutory basis …"); In re Fritch, 972 F.2d 1260, 1266, 23 U.S.P.Q.2d 1780, 1784 (Fed. Cir. 1992)("It is impermissible to use the claimed invention as an instruction manual or 'template' to piece together the teachings of the prior art so that the claimed invention is rendered obvious."); Orthopedic Equipment Co., Inc. v. United States, 702 F.2d 1005, 1012, 217 U.S.P.Q. 193, 199 (Fed.

Cir. 1983) (It is improper to attempt to establish obviousness by using the applicant's specification as a guide to combining different prior art references to achieve the results of the claimed invention.).

Claims 9 and 34

In paragraph 5 on page 9 of the Official Action dated Dec. 23, 2005, claims 9, 16-18, 34, 41-43, and 59-61 were rejected under 35 U.S.C. 103(a) as being unpatentable over Goldstein view of Ohran. This rejection was explicitly repeated for claims 9, 16, 34, 41, and 59 on page 6 of the Final Official Action dated May 31, 2006 and apparently also "sustained" for claims 17, 18, 42, 43, 60, and 61 on page 13 of the final Official Action. Appellants respectfully disagree and submit that that there is insufficient motivation in the prior art as a whole to combine Goldstein and Ohran in the fashion suggested in the Official Action, and the claimed invention would not result from a proper combination of Goldstein and Ohran.

Goldstein is summarized above with respect to applicants' claims 8, 33, and 54.

Ohran discloses a method of restoring a mass storage device, including the corresponding data blocks stored thereon, to a state in which it existed at a prior instant in time to minimize the data loss caused by data blocks becoming corrupt or lost. After a mirrored or backup copy has been made, data blocks that are to be overwritten in response to a write request are stored in a preservation memory prior to being overwritten. The data blocks stored in the preservation memory are time-stamped to designate the chronological order by which the data blocks were overwritten. If data becomes corrupted, the data blocks of the preservation memory are applied

to the corrupted data in reverse chronological order until such time that a valid, non-corrupted set of data is obtained. In this manner, data more recent than that associated with the full mirrored or backup copy can be reconstructed. (Ohran, Abstract.)

With respect to appellants' claims 9, 16, 41, and 59, page 7 of the final Official Action says: "Goldstein does not teach 'wherein the snapshot copy facility has a meta bit map for each snapshot copy for indicating blocks of data that are know[n] to be invalid in …"

Page 7 of the final Official Action cites Oran for "writing invalid or corrupted data to certain data blocks in the mass storage device (See paragraph [0015] and [Fig 3]) …" Ohran's paragraph 15 (emphasis added) says:

> [0015] In the event that certain data blocks in the mass storage unit device are lost or become corrupted, the data blocks stored in the preservation memory can be used to incrementally restore or reconstruct a valid set of data without reverting completely back to the data as it exists at time T_o. If, for example, invalid or corrupted data is written to certain data blocks in the mass storage device after time T_o, the original, valid data blocks are stored in a preservation memory as described above. Using the time stamps specifying the chronological sequence in which the data blocks stored in the preservation memory were overwritten in the mass storage device, the data blocks in the preservation memory are written to the current data stored in the mass storage device.

Page 7 of the final Official Action concludes that "it would have been obvious at the time of the invention for one of ordinary skill in the art to have modify Goldstein by teachings of

Ohran, wherein Ohran's teachings of separating invalid data, while a set of data is eventually used to reconstruct the invalid data, and been combined with Goldstein's method will enhance checking for change of valid data." Appellants respectfully disagree.

Appellants' claims 9 and 34 define that the snapshot copy facility has an index for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. Such an index for each snapshot copy is not suggested by a generalized teaching that invalid or corrupted data in mass storage (e.g., a production file system) can be corrected by over-writing them with a backup copy of good data (e.g., a snapshot copy).

In addition, the final Official Action refers to Goldstein "comparing the based state snapshot and a subsequent series of data volume snapshots to confirm changes, which can only be done if block is valid." However, there is nothing in Goldstein suggesting that determining a list of snapshot differences requires a determination that the blocks are valid or not in a younger snapshot. Nor is there an inherent need for a meta bit map in a snapshot copy facility, as shown by a comparison of appellants' FIG. 34 for the case of the snapshot copy facility in appellants' FIG. 25, which does not have a meta bit map, to appellants' FIGS. 39 and 40 for the case of the snapshot copy facility in appellants' FIG. 42, which has a meta bit map for each snapshot file system. A retrospective view of inherency is not a substitute for some teaching or suggestion which supports the selection and use of the various elements in the particular claimed combination. In re Rijckaert, 9 F.3d 1531, 1534, 28 U.S.P.Q.2d 1955-1957 (Fed. Cir. 1993)(optimal condition of matching signal time exactly to recording time is not "inherent" in the prior art); In re Newell, 891 F.2d 899, 901, 13 U.S.P.Q.2d 1248, 1250 (Fed. Cir. 1989)(no teaching or suggestion in the prior art

that the belt drive of Weiss should be applied to the capstan of an ANSI type of tape cartridge in the manner done by Newell).

In contrast to Goldstein and Ohran, the appellants teach that it is desirable to use a meta bit map so that if the original content of a block is invalid, it need not be saved for a snapshot when new content is written to the block. Not only is less storage used for the snapshot save volume, but also there can be a decrease in the access time for write access to the production file system. A write operation to an invalid block can be executed immediately, without the delay of saving the original contents of the data block to the most recent save volume at the tail of the snapshot queue. (Appellants' specification, page 69, line 8 to page 70, line 5.) In this case, however, a problem arises in that "the bit map for each snapshot (L) indicates whether or not a block has been stored in the save volume for the snapshot (L), and no longer will indicate all of the blocks that have been changed after snapshot (L) and before snapshot (L+1). … Therefore, for the preferred snapshot copy facility, it is desirable to modify the procedure of FIG. 34 in order to use the information in the meta bit map for the snapshot <M>. In this case, the procedure of FIG. 34 should also be modified to account for the fact that the save volumes no longer store the 'before images' for all of the blocks that may have changed between the successive snapshot volumes." (Appellants' specification, page 66, line 16 to age 67 line 10.)

Furthermore, it is not seen where Ohran teaches checking for blocks not known to be invalid before checking for changes. Instead, Ohran teaches restoring a set of invalid data blocks by incrementally applying data blocks in a preservation memory in reverse chronological order until such time that a valid set of data is obtained. In other words, given that some blocks in a

set have become invalid, Ohran restores the set of blocks by writing "before images" of the blocks in reverse chronological order. This is opposite from the appellants' determining whether a block has changed upon finding that the block is not known to be invalid. Appellants' claim 9, for example, does not recite restoring invalid data blocks. Instead, claim 9 specifically defines a snapshot copy facility responding to a request for the difference between a specified older snapshot copy and a specified younger snapshot copy. The snapshot copy facility responds to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies. The snapshot copy facility has an index for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy. The method includes scanning the index for the specified younger one of the snapshot copies, and when the index indicates that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies.

In short, responding to a request for the difference between a specified older snapshot copy and a specified younger snapshot copy by determining whether a block has changed between the specified older snapshot copy and a specified younger snapshot copy when an index indicates that the block is not known to be invalid, should not be confused with restoring potentially corrupted data blocks by incrementally applying "before-images" in reverse chronological order until the data is rolled-back to a valid state.

Hindsight reconstruction, using the applicant's specification itself as a guide, is improper because it fails to consider the subject matter of the invention "as a whole" and fails to consider the

invention as of the date at which the invention was made. "[T]here must be some motivation, suggestion, or teaching of the desirability of making the specific combination that was made by the applicant." In re Lee, 277 F.3d 1338, 1343, 61 U.S.P.Q.2d 1430, 1435 (Fed. Cir. 2002) (quoting In re Dance, 160 F.3d 1339, 1343, 48 U.S.P.Q.2d 1635, 1637 (Fed. Cir. 1998)). "[T]eachings of references can be combined only if there is some suggestion or incentive to do so." In re Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988) (Emphasis in original) (quoting ACS Hosp. Sys., Inc. v. Montefiore Hosp., 732 F.2d 1572, 1577, 221 U.S.P.Q. 929, 933 (Fed. Cir. 1984)). "[P]articular findings must be made as to the reason the skilled artisan, with no knowledge of the claimed invention, would have selected these components for combination in the manner claimed." In re Kotzab, 217 F.3d 1365, 1371, 55 U.S.P.Q.2d 1313, 1317 (Fed. Cir. 2000). See, for example, Fromson v. Advance Offset Plate, Inc., 755 F.2d 1549, 1556, 225 U.S.P.Q. 26, 31 (Fed. Cir. 1985) (nothing of record plainly indicated that it would have been obvious to combine previously separate lithography steps into one process); In re Gordon et al., 733 F.2d 900, 902, 221 U.S.P.Q. 1125, 1127 (Fed. Cir. 1984) (mere fact that prior art could be modified by turning apparatus upside down does not make modification obvious unless prior art suggests desirability of modification); Ex Parte Kaiser, 194 U.S.P.Q. 47, 48 (PTO Bd. of Appeals 1975) (Examiner's failure to indicate anywhere in the record his reason for finding alteration of reference to be obvious militates against rejection).

Claims 16, 41, and 59.

In the final Official Action, claims 16, 41, and 59 were rejected on the grounds corresponding to the argument in the final Official Action given for rejecting claim 9. Claims 16, 41, and 59 are patentable over Goldstein and Ohran for the same reasons given above for claim 9. In addition, claims 16, 41, and 59 further define that the sequence of the snapshot copies includes the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies, and the indices for the sequence of the snapshot copies are scanned by a program routine having an outer loop indexing respective blocks, and an inner loop indexing snapshot copies in the sequence of the snapshot copies. Thus, claims 16, 41, and 59 further distinguish Goldstein for the reasons given above with respect to claims 8, 33, and 54.

Claims 17, 42, and 60.

With respect to appellants' claims 17, 18, 42, 43, 60, and 61, page 10 of the Official Action of Dec 23, 2005, makes a similar argument with respect to the proposed combination of Goldstein and Ohran for the rejection of claim 9. Therefore, claims 17, 42, and 60 (and their dependent claims 18, 43, and 61) are patentable for the reasons given above for claim 9. In addition, claims 17, 42, and 60 specify a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system and that have a "before image" saved for said each snapshot copy, and a second index for said each snapshot copy for indicating blocks of data

that are not in use in said each snapshot copy. The Official Action says: "Ohran teaching include writing invalid or corrupted data to a certain block. Wherein the abstract states data loss could be caused by data blocks becoming corrupt or lost, therefore data not in use is equivalent to data loss." The appellants respectfully disagree. In the context of Ohran's reconstruction of corrupted data, it is not understood how one can conclude from the abstract of Ohran that data not in use is equivalent to data loss. There is no need to reconstruct data not in use, but there is a need to reconstruct corrupted data.

Claims 18, 43, and 61

Claims 18, 43, and 61 are dependent on claims 17, 42, and 60, and are therefore patentable over Goldstein and Ohran for the reasons given above with respect to claims 17, 42, and 60. In addition, claims 18, 43, and 61 define "accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the sequence of the snapshot copies to determine that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies not changed." For example, determining that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies occurs in steps 656 and 657 of appellants' FIG. 39, and more specifically in the inner loop of steps 667, 668, 669, and 670 in FIG. 40. The "accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the

sequence of the snapshot copies to determine that said at least one of the blocks has changed" occurs in step 668 of FIG. 40 when the meta bit map (I, Bi) is not equal to one, causing execution to branch from step 668 to step 671 in FIG. 40. This accessing of the at least one of the second indices for the snapshot copies is done "to account for the fact that the save volumes no longer save the "before images" for all of the blocks that may have changed between the successive snapshot copies." (Appellants' specification, page 67, lines 7 to 10.) It is not understood how the further limitations of claims 18, 43, and 61 would have been suggested by Goldstein or Ohran.

In view of the above, the rejection of the appellants' claims should be reversed.

Respectfully submitted,

Richard C. Auchterlonie
Reg. No. 30,607

NOVAK DRUCE & QUIGG, LLP
1000 Louisiana, 53rd Floor
Houston, TX 77002
713-571-3460

VIII. CLAIMS APPENDIX

8. A method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, said method comprising:

the snapshot copy facility receiving a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies; and

the snapshot copy facility responding to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies;

wherein the snapshot copy facility has an index for each snapshot copy for indicating changes between said each snapshot copy and a next snapshot copy of the production file system, and the method includes scanning the index for the specified older one of the snapshot copies,

which includes scanning the indices for a sequence of the snapshot copies including the index for the specified older one of the snapshot copies and a respective index for each of a plurality of snapshot copies of the production file system that are both younger than the specified older one snapshot copies and older than the specified younger one of the snapshot copies, and

wherein the indices for the sequence of the snapshot copies are scanned by a program routine having an outer loop indexing blocks of data in the file system, and an inner loop indexing the snapshot copies in the sequence of the snapshot copies.

9. A method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, said method comprising:

the snapshot copy facility receiving a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies; and

the snapshot copy facility responding to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies;

wherein the snapshot copy facility has an index for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy, and the method includes scanning the index for the specified younger one of the snapshot copies, and when the index indicates that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies.

16. A method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, the snapshot copy facility having an index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system, wherein the method comprises:

scanning the indices for a sequence of the snapshot copies to determine the blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies;

wherein the indices for the sequence of the snapshot copies are scanned by a program routine having an outer loop indexing respective blocks, and an inner loop indexing snapshot copies in the sequence of the snapshot copies; and

wherein the snapshot copy facility has a meta bit map for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy, and the method includes scanning the meta bit map for the specified younger one of the snapshot copies, and when the meta bit map is found to indicate that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies by scanning the indices for the sequence of the snapshot copies.

17. A method of operating a snapshot copy facility that stores a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, the snapshot copy facility having a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system

and that have a "before image" saved for said each snapshot copy, the snapshot copy facility having a second index for said each snapshot copy for indicating blocks of data that are not in use in said each snapshot copy; said method comprising:

responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by accessing the second index for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices for a sequence of the snapshot copies to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies.

18. The method as claimed in claim 17, which also includes accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the sequence of the snapshot copies to determine that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies not changed.

33. A snapshot copy facility comprising:

storage for storing a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time; and

at least one processor programmed for receiving a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies; and for responding to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies;

wherein the snapshot copy facility has an index for each snapshot copy for indicating changes between said each snapshot copy and a next snapshot copy of the production file system, and said at least one processor is programmed for scanning the index for the specified older one of the snapshot copies,

wherein said at least one processor is programmed for scanning the indices for a sequence of the snapshot copies including the index for the specified older one of the snapshot copies and a respective index for each of a plurality of snapshot copies of the production file system that are both younger than the specified older one snapshot copies and older than the specified younger one of the snapshot copies, and

wherein said at least one processor is programmed for scanning the indices for the sequence of the snapshot copies by a program routine having an outer loop indexing the blocks, and an inner loop indexing the snapshot copies in the sequence of the snapshot copies.

34. A snapshot copy facility comprising:

storage for storing a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time; and

at least one processor programmed for receiving a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies; and for responding to the request by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies;

wherein the snapshot copy facility has an index for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy, and said at least one processor is programmed for scanning the index for the specified younger one of the snapshot copies, and when the index indicates that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies.

41. A snapshot copy facility comprising:

storage for storing a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time;

an index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system, and

at least one processor programmed for scanning the indices for a sequence of the snapshot copies to determine the blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies,

which includes a meta bit map for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy, and wherein said at least one processor is programmed for scanning the meta bit map for the specified younger one of the snapshot copies, and when the meta bit map is found to indicate that a block is not known to be invalid, then determining whether the block has changed between the specified older one of the snapshot copies and the specified younger one of the snapshot copies by scanning the indices for the sequence of the snapshot copies.

42. A snapshot copy facility comprising:

storage for storing a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time;

a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system and that have a "before image" for said each snapshot copy stored in the storage,

a second index for each snapshot copy for indicating blocks of data that are not in use in said each snapshot copy, and

at least one processor programmed for responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by accessing the second index for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices for a sequence of the snapshot copies to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies.

43. The snapshot copy facility as claimed in claim 42, wherein said at least one processor is also programmed for accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the sequence of the snapshot copies to determine that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies not changed.

54. A program storage device containing a program for a snapshot copy facility, the snapshot copy facility storing a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, wherein the program is executable for responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by returning the difference between the specified older one of the snapshot copies and the specified younger one of the snapshot copies,

wherein the snapshot copy facility has an index for each snapshot copy for indicating changes between said each snapshot copy and a next snapshot copy of the production file system, and the program is executable for scanning the index for the specified older one of the snapshot copies,

wherein the program is executable for scanning the indices for a sequence of the snapshot copies including the index for the specified older one of the snapshot copies and a respective index for each of a plurality of snapshot copies of the production file system that are both younger than the specified older one snapshot copies and older than the specified younger one of the snapshot copies,

wherein the program is executable for scanning the indices for a sequence of the snapshot copies including the index for the specified older one of the snapshot copies and a respective index for each of a plurality of snapshot copies of the production file system that are both

younger than the specified older one snapshot copies and older than the specified younger one of the snapshot copies, and

wherein the program is executable for scanning the indices for the sequence of the snapshot copies by a program routine having an outer loop indexing the blocks, and an inner loop indexing the snapshot copies in the sequence of the snapshot copies.

59. A program storage device containing a program for a snapshot copy facility, the snapshot copy facility having a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, and an index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system, wherein the program is executable for scanning the indices for a sequence of the snapshot copies to determine the blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies,

wherein the snapshot copy facility has a meta bit map for each snapshot copy for indicating blocks of data that are known to be invalid in said each snapshot copy, and wherein the program storage device is executable for scanning the meta bit map for the specified younger one of the snapshot copies, and when the meta bit map is found to indicate that a block is not known to be invalid, then determining whether the block has changed between the specified

older one of the snapshot copies and the specified younger one of the snapshot copies by scanning the indices for the sequence of the snapshot copies.

60. A program storage device containing a program for a snapshot copy facility, the snapshot copy facility having a plurality of snapshot copies of a production file system, each of the snapshot copies being a prior state of the production file system at a respective point in time, a first index for each snapshot copy for indicating blocks of data in the production file system that have changed between said each snapshot copy and a next snapshot copy of the production file system and that have a "before image" for said each snapshot copy stored in the snapshot copy facility, and a second index for each snapshot copy for indicating blocks of data that are not in use in said each snapshot copy, wherein the program is executable for responding to a request for the difference between a specified older one of the snapshot copies and a specified younger one of the snapshot copies by accessing the second index for the specified younger one of the snapshot copies to determine blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, and for blocks of data in the production file system that are in use in the specified younger one of the snapshot copies, accessing at least one of the first indices for a sequence of the snapshot copies to determine blocks that have changed between an older one of the snapshot copies and a younger one of the snapshot copies, the sequence of the snapshot copies including the older one of the snapshot copies and each of the snapshot copies that is both younger than the older one of the snapshot copies and older than the younger one of the snapshot copies.

61. The program storage device as claimed in claim 60, wherein the program is executable for accessing at least one of the second indices for the snapshot copies in the sequence of the snapshot copies and finding that at least one of the blocks is not in use in at least one of the snapshot copies in the sequence of the snapshot copies to determine that said at least one of the blocks has changed between the older one of the snapshot copies and the younger one of the snapshot copies not changed.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.